USB Flash Disk and SD Card File Management Control Chip

CH376

Datasheet (II): Auxiliary Commands and USB Basic Transmission Commands

Version: 1

http://wch.cn

1. Additional Commands

Code	Command name CMD_	Input data	Output data	Command purpose
04H	SET_USB_SPEED	Bus speed		Set USB bus speed
0AH	GET_DEV_RATE	Data 07H	Data rate type	Get the data rate type of the USB device
0AH	READ_VAR8	Variable address	Data	Read the specified 8-bit file system variables
0BH	SET_RETRY	Data 25H Number of retries		Set the number of retries for USB transaction operations
0BH	WRITE_VAR8	Variable address Data		Set the specified 8-bit file system variables
0CH	READ_VAR32	Variable address	Data (4 bytes)	Read the specified 32-bit file system variables
0DH	WRITE_VAR32	Variable address Data (4 bytes)		Set the specified 32-bit file system variables
0FH	DELAY_100US		Delay status	Delay 100uS
13H	SET_USB_ADDR	Address value		Set USB address
16H	TEST_CONNECT		(Wait for 2uS) Connection status	Check the connection status of USB device
17H	ABORT_NAK			Abort the retry of the current NAK.
1CH	SET_ENDP6	Working mode	(Wait for 3uS)	Set the receiver for USB host endpoint
1DH	SET_ENDP7	Working mode	(Wait for 3uS)	Set the transmitter for USB host endpoint
25H	DIRTY_BUFFER			Clear internal disk and file buffers
41H	CLR_STALL	Endpoint number	Generate interrupt	Control transmission: Clear endpoint error
45H	SET_ADDRESS	Address value	Generate interrupt	Control transmission: Set USB address
46H	GET_DESCR	Descriptor type	Generate interrupt	Control transmission: Get the descriptor
49H	SET_CONFIG	Configuration Value	Generate interrupt	Control transmission: Set USB configuration
4DH	AUTO_SETUP		Generate interrupt	Automatically configure USB device
4EH	ISSUE_TKN_X	Sync flag Transaction attribute	Generate interrupt	Send synchronization token, and execute transactions

51H	DISK_INIT		Generate interrupt	Initialize USB memory
52H	DISK_RESET		Generate interrupt	Reset USB storage device
53H	DISK_SIZE		Generate interrupt	Get the maximum capacity of the USB memory
58H	DISK_INQUIRY		Generate interrupt	Query USB memory features
59H	DISK_READY		Generate interrupt	Check USB memory readiness
5AH	DISK_R_SENSE		Generate interrupt	Check USB memory error
5BH	RD_DISK_SEC		Generate interrupt	Read sector data from the disk to the internal buffer
5CH	WR_DISK_SEC		Generate interrupt	Write sector data from the internal buffer to the disk
5DH	DISK_MAX_LUN		Generate interrupt	Get the maximum unit number of the USB storage device

1.1. Command CMD_SET_USB_SPEED

This command is used to set USB bus speed. This command needs the input of 1 data to select the USB bus speed. 00H corresponds to 12Mbps full speed mode, 01H corresponds to 1.5MBps full speed mode (non-standard mode), and 02H corresponds to 1.5Mbps low speed mode. The USB bus speed of CH376 is 12Mbps full speed by default, and will be restored to 12Mbps full speed mode after the command CMD_SET_USB_MODE is executed to set USB working mode.

1.2. Command CMD_GET_DEV_RATE

This command is used to get the data rate type of the currently connected USB device. This command requires to input one data 07H, output the data rate type. If the bit 4 is 1, the device will be 1.5Mbps low speed USB device; otherwise, it will be 12Mbps full speed USB device. This command is only valid in USB mode 5 (enabled USB host mode, not generating SOF package).

1.3. Command CMD_READ_VAR8

This command is used to read the specified 8-bit (single byte) file system variables. This command requires to input 1 data specified variable address and output the data for this variable.

1.4. Command CMD_SET_RETRY

This command is used to set the number of retries for USB transaction operations. This command requires to input two data, respectively the data 25H and the number of retries.

The bits 7 and 6 of the number of retries specify the processing mode when CH376 receives NAK response. If the bit 7 is 1 and the bit 6 is 0, infinite retry will be performed (the current retry can be given up through the command CMD_ABORT_NAK); if the bit 7 is 1 and the bit 6 is 1, finite retry will be performed for 3 seconds around; if the bit 7 is 0, NAK will be notified to MCU as the result or processed as an error. Bits 5-0 of the number of retries specify the number of retries CH376 after the USB device response timeout. If the bit is 0, retry will not be performed after timeout.

The default number of retries is 8FH after chip reset or USB mode reset, so infinite retry will be performed after NAK reply, and the response is received, and 15 retries will be performed after USB device response timeout.

1.5. Command CMD_WRITE_VAR8

This command is used to set the specified 8-bit (single byte) file system variables. This command requires to input two data, respectively the specified variable address and the specified variable data.

1.6. Command CMD_READ_VAR32

This command is used to read the specified 32-bit (4 bytes) file system variables. This command requires to input 1 data specified variable address and output the data for this variable. The variable data has a total of 4 bytes, which are the lowest byte of data, the lower byte of data, the higher byte of data, and the highest byte of data.

1.7. Command CMD_WRITE_VAR32

This command is used to set the specified 32-bit (4 bytes) file system variables. This command requires to input 5 data, respectively specified variable address, the lowest byte of variable data, lower byte of data, higher byte of data and the highest byte of data.

1.8. Command CMD_DELAY_100US

This command is used to delay for 100uS and does not support serial port mode. The output data is 0 during delay, and is non-0 (usually the chip version number) at the end of the delay. MCU determines whether the delay is ended according to the read data.

1.9. Command CMD_SET_USB_ADDR

This command is used to set the USB device address. This command requires to input 1 data to select the address of the USB device being operated. After reset or after the USB device is connected or disconnected, the USB device address is always 00H, and MCU communicates with the USB device through the default address 00H. If MCU sets the address of the USB device through the standard USB request, it must set the same USB device address through the command, so that CH376 can communicate with the USB device through the new address.

1.10. Command CMD_TEST_CONNECT

This command is used to query the connection status of the current USB device in USB host mode. Typically, this command is completed within 2uS. After completion, USB_INT_CONNECT, USB_INT_DISCONNECT, or USB_INT_USB_READY is output. USB_INT_CONNECT indicates that the USB device is just connected or has been connected but has not been initialized. USB_INT_DISCONNECT indicates that the USB device has not been connected or has been disconnected. Status USB_INT_USB_READY indicates that USB device has been connected and initialized (USB address has been assigned). 0 indicates that the command is not completed and the status can be read later.

1.11. Command CMD_ABORT_NAK

This command is used to abort the retry of the current NAK. When CH376 works in the USB host mode, by default, it will keep retrying until it returns success or error when receiving the NAK status returned by the USB device. This command can force CH376 to terminate a retry in order to perform a new operation. In addition, the command SET_RETRY can be used to set whether NAK retry is disabled.

1.12. Command CMD_SET_ENDP6

This command is used to set either the USB host endpoint or the receiver of the endpoint 2 (OUT of device mode endpoint 2 / IN of host endpoint). This command needs to input 1 data to specify a new working mode. For example, if an IN transaction is executed, DATA0 is expected to be received and DATA1 is given up, the synchronous trigger flag of the receiver at the host endpoint must be set to 0 through this command. The byte in the corresponding working mode byte is 80H. Typically, this command is completed within 3uS.

For the working mode of the transceiver for USB endpoint, refer to the following table.

Working mode byte	Name	Bit analysis description of working mode	
Bits 7-6	Synchronous trigger flag	If the bit 7 is 1, the bit 6 will be the new synchronous trigger flag:	
		00 or 01 = Keep the current synchronous trigger flag unchanged	
		10 = Synchronous trigger flag set to 0	11 = Synchronous trigger flag set to 1
Bits 5-4	(Reserved bit)	(Undefined, must be 0)	
Bits 3-0	Transaction response mode	Must be 0000	

1.13. Command CMD_SET_ENDP7

This command is used to set either the USB host endpoint or the transmitter of the endpoint 2 (IN of device mode endpoint 2 / OUT of host endpoint). This command needs to input 1 data to specify a new working mode. For example, if SETUP or OUT transaction is executed and DATA0 is expected to be transmitted, the synchronous trigger flag of the transmitter at the host endpoint must be set to 0 through this command. The corresponding working mode byte is 80H. If DATA1 is expected to be transmitted, the working mode byte will be C0H. Typically, this command is completed within 3uS.

1.14. Command CMD_DIRTY_BUFFER

This command is used to clear internal disk and file buffers in the host file mode. After being in host file mode, CH376 always stores some frequently used data in the internal disk buffer, but when some commands (such as CMD_RD_DISK_SEC or CMD_WR_DISK_SEC, etc.) are executed, the buffer will be used, causing the buffer data to be invalid. In order to prevent CH376 from misusing the invalid data, it is necessary to notify CH376 to clear the internal buffer after other commands use the internal buffer.

1.15. Command CMD_CLR_STALL

This command is used to clear the wrong control transmission command of the endpoint. This command requires to input 1 data to specify the endpoint address of the USB device for which errors will be cleared. Valid addresses are 01H ~ 0FH for OUT endpoint and 81H ~ 8FH for IN endpoint. This command is used to simplify the standard USB request CLEAR_FEATURE. CH376 requests an interrupt from MCU after the command is executed. If the interrupt status is USB_INT_SUCCESS, the command will be executed successfully.

1.16. Command CMD_SET_ADDRESS

This command is used to set the control transmission command of the USB address. This command requires to input 1 data to specify the new address USB device, and the valid addresses are 00H-7FH. This command is used to simplify the standard USB request SET_ADDRESS. CH376 requests an interrupt from MCU after the command is executed. If the interrupt status is USB_INT_SUCCESS, the command will be executed successfully.

1.17. Command CMD_GET_DESCR

This command is used to get the control transmission command of the descriptor. This command requires to input 1 data to specify the type of descriptor to be obtained. The valid type is 1 or 2, corresponding to DEVICE descriptor and CONFIGURATION descriptor. CONFIGURATION descriptor also includes the interface descriptor and the endpoint descriptor. This command is used to simplify the standard USB request GET_DESCRIPTOR. CH376 requests an interrupt from MCU after the command is executed. If the

interrupt status is USB_INT_SUCCESS, the command will be executed successfully. MCU can get the descriptor data through the command CMD_RD_USB_DATA0. As the control transmission buffer of CH376 is only 64 bytes, CH376 will return the operation status USB_INT_BUF_OVER when the length of the descriptor exceeds 64 bytes. For this USB device, MCU can process the control transmission through the command CMD_ISSUE_TKN_X.

1.18. Command CMD_SET_CONFIG

This command is used to set the control transmission command of the USB configuration. This command requires to input 1 data to specify a new USB configuration value. If the configuration value is 0, the configuration will be canceled, otherwise it shall be taken from the configuration descriptor of the USB device. This command is used to simplify the standard USB request SET_CONFIGURATION. CH376 requests an interrupt from MCU after the command is executed. If the interrupt status is USB_INT_SUCCESS, the command will be executed successfully.

1.19. Command CMD_AUTO_SETUP

This command is used to automatically configure USB device and does not support SD card. This command is used to simplify the initialization steps for the common USB device and is equivalent to the sequence of multiple commands such as GET_DESCR, SET_ADDRESS, SET_CONFIGURATION and so on. CH376 requests an interrupt from MCU after the command is executed. If the interrupt status is USB_INT_SUCCESS, the command will be executed successfully.

1.20. Command CMD_ISSUE_TKN_X

This command enables CH376 to send a synchronous token to execute the transaction. This command requires to input two data, respectively synchronization flag and transaction attribute. The bit 7 of the synchronization flag is the synchronous trigger flag of the receiver at the host endpoint, the bit 6 is the synchronous trigger flag of the transmitter at the host endpoint, and the bits 5-0 must be 0. The low 4 bits of the transaction attribute specify the token PID of the transaction, and the high 4 bits specify the destination endpoint number of the USB device. CH376 requests an interrupt from MCU after the command is executed. MCU can read the interrupt status as the operation status of the command. If the operation status is USB_INT_SUCCESS, the command will be executed successfully; otherwise, the command will be executed unsuccessfully. MCU can further analyze the cause of failure according to the operation status.

For SETUP and OUT transactions sending data, first write the data to be sent through the command CMD_WR_HOST_DATA, and then execute the transaction through the command CMD_ISSUE_TKN_X. For IN transaction receiving data, first execute the transaction through the command CMD_ISSUE_TKN_X and read the received data through the command CMD_RD_USB_DATA0 after successful execution.

The following table shows USB token PID supported by CH376.

PID byte	Name	Description
0DH	DEF_USB_PID_SETUP	Initiate control transmission, and send setup data
01H	DEF_USB_PID_OUT	Execute OUT transaction, and send data
09H	DEF_USB_PID_IN	Execute IN transaction, and receive data

For example, when the transaction attribute byte is 09H, CH376 receives data from the default endpoint 0 of the USB device; when the transaction attribute byte is 21H, CH376 sends data to the endpoint 2 of the USB device; when the transaction attribute byte is 29H, CH376 receives data from the endpoint 2 of the USB device, with address of 82H.

1.21. Command CMD_DISK_INIT

This command is used to initialize USB storage device and does not support SD card. For the connected USB device, the command first resets the USB bus, then analyzes the descriptor of the USB device. If it is a

supported USB storage device, the device will be automatically configured, and a connection to the USB storage device will be finally established. CH376 requests an interrupt from MCU after the command is executed. If the USB device has been disconnected, the interrupt status may be USB_INT_DISCONNECT. If the USB device is not recognized or the USB storage device is not supported, the interrupt status will be usually USB_INT_DISK_ERR or USB_INT_BUF_OVER. If the USB storage device is initialized successfully, the interrupt status will be USB_INT_SUCCESS.

1.22. Command CMD_DISK_RESET

This command is used to reset USB storage device through the control transmission and does not support SD card. CH376 requests an interrupt from MCU after the command is executed. If the interrupt status is USB_INT_SUCCESS, the command will be executed successfully.

The complete reset process includes: resetting the USB storage device through this command, resetting Bulk-IN endpoint through the command CLR_STALL, and resetting Bulk-OUT endpoint through the command CLR_STALL.

When an error occurs on the USB storage device, CH376 will analyze the cause of the error and automatically select whether the USB device is reset or not as required.

1.23. Command CMD_DISK_SIZE

This command is used to get the capacity of the USB storage device and does not support SD card. After successfully initializing the USB storage device, this command gets the total capacity of the USB storage device. CH376 requests an interrupt from MCU after the command is executed. If the interrupt status is USB_INT_SUCCESS, the data can be gotten through the command CMD_RD_USB_DATA0. The data is usually 8 bytes. The first 4 bytes constitute double-word data with high bytes in the front, which is the total number of sectors of USB storage device. The last 4 bytes constitute the double-word data with high bytes in the front, which is the number of bytes of each sector. The result of multiplying two data is the total capacity of USB storage device in bytes.

1.24. Command CMD_DISK_INQUIRY

This command is used to inquire the features of the USB storage device and does not support SD card. CH376 requests an interrupt from MCU after the command is executed. If the interrupt status is USB_INT_SUCCESS, the data can be gotten through the command CMD_RD_USB_DATA0. The data is usually 36 bytes, including the features of USB storage device and the identification information of manufacturer and product. This command is generally not needed unless a new logical unit is analyzed.

1.25. Command CMD_DISK_READY

This command is used to check whether the USB storage device is ready and does not support SD card. CH376 requests an interrupt from MCU after the command is executed. If the interrupt status is USB_INT_SUCCESS, it will indicate that the USB storage device has been ready.

1.26. Command CMD_DISK_R_SENSE

This command is used to check the error of the USB storage device. CH376 requests an interrupt from MCU after the command is executed. The interrupt status is normally USB_INT_SUCCESS, and the error can be analyzed after the data is gotten through the command CMD_RD_USB_DATA0.

1.27. Command CMD_RD_DISK_SEC

This command is used to read data from a sector from the disk to the internal buffer in the host file mode.

1.28. Command CMD_WR_DISK_SEC

This command is used to write data for a sector of the internal buffer to the disk in the host file mode.

1.29. Command CMD_DISK_MAX_LUN

This command is used to get the maximum logical unit number of the USB storage device through control transmission. CH376 requests an interrupt from MCU after the command is executed. If the interrupt status is USB_INT_SUCCESS, the data can be gotten through the command CMD_RD_USB_DATA0. The data is usually 1 byte.

2. External Firmware

2.1. Overview

The command CMD_ISSUE_TKN_X is used to perform basic USB transmission transactions. It is the most basic operation in firmware programming in the USB host mode. On this basis, the external MCU can process the control transmission for which CH376 has not directly provided the simplified commands according to the requirements of USB protocol. Further, MCU can process the USB protocol of a variety of specific types of devices in accordance with the requirements of the USB protocol, to achieve the control and data exchange of USB devices. CH376 has Bulk-Only transport protocol for Mass-Storage devices. For USB Storage devices using CBI transport protocol or other transport protocols, the external MCU still needs to process based on the command CMD_ISSUE_TKN_X and the control transmission command.

2.2. External Firmware Reference Flow

The external firmware reference program is provided in the CH376 evaluation board data. The following process is that the external MCU executes the standard USB request GET_STATUS through the control transmission to get the status of USB device for reference when the external MCU designs the firmware program.

(1) Setup stage of control transmission

① Send the command CMD_WR_HOST_DATA to write 8 bytes of request data to the output buffer. The request data is 80H, 00H, 00H, 00H, 00H, 02H and 00H in order, and the length is 8.

② Send the command CMD_ISSUE_TKN_X to execute a transaction with the synchronization flag of 00H and the transaction attribute byte of 0DH, send a SETUP token to the default endpoint 0, and send DATA0.

③ MCU waits for the completion of transaction interrupt or wait for the interrupt notification.

④ After the transaction is completed, CH376 sets INT# pin to low level and requests an interrupt from MCU.

⑤ MCU enters the interrupt service program, or exits after receiving the interrupt notification in the main program.

⑥ Send the command CMD_GET_STATUS to get the interrupt status.

⑦ CH376 restores INT# pin to a high level and cancels the interrupt request after the command CMD_GET_STATUS is completed..

⑧ MCU analyzes the obtained interrupt status. If it is not USB_INT_SUCCESS, the operation will fail and exception will be handled; if it is USB_INT_SUCCESS, the transaction will be executed successfully and the setup stage will be completed.

(2) Data stage of control transmission

① Send the command CMD_ISSUE_TKN_X to execute a transaction with the synchronization flag of 80H and the transaction attribute byte of 09H, send an IN token to the default endpoint 0, and send DATA1.

② MCU waits for the transaction to be completed; after the transaction is completed, CH376 requests an interrupt from MCU.

③ Send the command CMD_GET_STATUS to get the interrupt status. CH376 cancels the interrupt request.

④ MCU analyzes the interrupt status. If the operation failed, exception will be handled; if it is USB_INT_SUCCESS, the transaction will be executed successfully.

⑤ Send the command CMD_RD_USB_DATA0 to get the data returned by the USB device and save it as the return result of the control transmission.

⑥ Because this control transmission requires only one IN transaction, the data stage is completed.

(3) Status stage of control transmission

① Send the command CMD_WR_HOST_DATA to write the status data of the length 0 to the output buffer. The length is 0.

② Send the command CMD_ISSUE_TKN_X to execute a transaction with the synchronization flag of 40H and the transaction attribute byte of 01H, send an OUT token to the default endpoint 0 and send DATA1.

③ MCU waits for the transaction to be completed; after the transaction is completed, CH376 requests an interrupt from MCU.

④ Send the command CMD_GET_STATUS to get the interrupt status. CH376 cancels the interrupt request.

⑤ MCU analyzes the obtained interrupt status. If the operation failed, exception will be handled; if it is USB_INT_SUCCESS, the transaction will be executed successfully and the status stage will be completed.

(4) The control transmission is completed. The data returned in the data stage is returned as the standard USB request GET_STATUS. Typically, the length of returned data is 2 bytes.